1. **Что такое COM? COM-программирование?**

Common Object Model. Модель программного обеспечения, которая разработана Microsoft (может выполняться в любой ОС). Версия из книги: спецификация, указывающая как создавать динамически взаимозаменяемые компоненты.

COM — это технология, которая позволяет объектам взаимодействовать между процессами и границами компьютеров так же легко, как в рамках одного процесса.

COM – программирование – разработка программного обеспечения используя модель COM.

1. **Что такое COM-объект(компонент)? CLSID?**

**COM-объект** - специализированный объект времени исполнения (экземпляр).

**CLSID** — это глобальный уникальный идентификатор, идентифицирующий объект класса COM. Если сервер или контейнер позволяют связываться со встроенными объектами, необходимо зарегистрировать CLSID для каждого поддерживаемого класса объектов.

1. **Что такое GUID? Где применяется GUID? Размер GUID-идентификатора?**

**GUID** – тип идентификатора объекта.

Используется, как уникальный идентификатор COM-объектов.

Размер 128 bit.

1. **Какие типы COM-контейнеров бывают?**

**EXE, DLL**

1. **Что является клиентом и сервером в COM?**

Сервер – программный модуль, реализующий COM-объект

Клиент – программный модуль, который создает объект и использует его.

1. **Поясните понятия «однокомпонентный» и «многокомпонентный» COM-сервер.**

однокомпонентным (реализующим один тип объектов) или многокомпонентным (реализующим несколько типов объектов).

1. **Поясните типы COM-серверов: CLSCTX\_INPROC\_SERVER, CLSCTX\_LOCAL\_SERVER, СLSCTX\_REMOTE\_SERVER.**

CLSCTX\_INPROC\_SERVER (DLL внутрипроцессный сервер); CLSCTX\_LOCAL\_SERVER (EXE-сервер за границами процесса, но та том же компьютере); СLSCTX\_REMOTE\_SERVER (EXE-сервер на удаленном компьютере)

1. **Как называется имя библиотеки, обеспечивающей работу COM-приложений.**

OLE32.DLL (и библиотека импорта OLE32.LIB)

1. **Поясните назначение типа и структуру HRESULT.**

Большинство методов СОМ-интерфейса и функции API возвращают значение типа HRESULT (исключениями являются AddRef и Release). В Win32 тип данных HRESULT определяется как DWORD (32-битовое целое), а возвращаемое значение этого типа содержит информацию о результате вызова функции. Старший бит сигнализирует об успешном или ошибочном завершении работы функции, а следующие 15 битов идентифицируют тип ошибки и обеспечивают способ группировки однотипных кодов завершения; младшие 16 битов предоставляют специфическую информацию о происшедшем. Структура данных HRESULT идентична структуре значений флагов статуса, используемых функциями Win32 API.

1. **Что такое COM-интерфейс?**

Применяется для доступа к методам COM-объекта.

1. **Чем характеризуется COM-интерфейс?**

Каждый интерфейс включает один или несколько методов, имеет идентификатор GUID,

1. **Что значит «стандартный» COM-интерфейс?**

Интерфейс, который описан в спецификации COM

1. **Назовите два стандартных COM-интерфейса.**

* IUnknown (QueryInterface, AddRef, Realese)
* IClassFactory (CreateInstance, LockServer) **наследуется от IUnknown**

1. **Перечислите методы интерфейса IUnknown и поясните их назначение.**

* QueryInterface (находит интерфейсы по IID (При вызове метода QueryInterface необходимо использовать идентификатор интерфейса (IID). IID — это глобально уникальное значение идентификатора (GUID).))
* AddRef (увеличивает счетчик ссылок на интерфейс (компонент) на единицу)
* Realese (уменьшает счетчик ссылок на интерфейс (компонент) на единицу)

1. **Что такое «фабрика классов» и для чего она нужна?**

Стандартный компонент, который обязан реализовывать интерфейс IClassFactory, создающий экземпляры пользовательских компонентов

1. **Перечислите методы интерфейса IClassFactory и поясните их назначение.**

* CreateInstance (возвращает экземпляр пользовательского компонента)
* LockServer (запрещает разрушение экземпляра фабрики классов, имеет смысл когда мы хотим создать несколько экземпляров пользовательского компонента)

1. **Что такое «счетчик ссылок на интерфейсы»? Для чего он нужен? Каким образом и когда этот счетчик увеличивается и уменьшается?**

Данный механизм позволяет узнать, когда можно удалить компонент из памяти.   
AddRef (увеличивает счетчик)

Realese (уменьшает счетчик)

При вызове QueryInterface (AddRef должен вызываться автоматически)

Клиент обязан вызывать AddRef только в случае копирования ссылки на интерфейс (компонент)

1. **Какое соглашение о вызове и возврате должен обеспечивать метод COM-объекта? Какие методы являются исключением?**

Все методы COM-объекта должны возвращать HRESULT (исключениями являются AddRef и Release). Для того чтобы получить результат работы функции необходимо пользоваться выходными параметрами.

1. **Что должен «знать» COM-клиент, чтобы использовать COM-объект?**

для создания COM-объекта и работы с ним должен знать: 1)CLSID объекта; 2) тип DLL-сервера (контейнера); 3) ID интерфейсов объекта.

1. **Объясните в чем заключается процесс регистрации COM-объекта?**

Поместить в реестер имена файлов компонента, индексированные CLSID

1. **Поясните назначение утилиты regsvr32 и принцип ее работы.**

Позволяет выполнить регистрацию и отмену регистрации DLL в реестре ОС

1. **Поясните назначение утилиты regedit.**

Позволяет просматривать и изменять реестер

1. **Перечислите пять функций, которые экспортируются COM/DLL-контейнером. Поясните назначение этих функций.**

DllRegisterServer – используется revsvr32 для саморегистрации компонента

DllUnregisterServer – также, только для самоотрегистрации

DllInstall – используется regsvr32 чтобы понять можно ли ей что-угодно делать (вызвать DllRegisterServer, например)

DllCanUnloadNow – юзает OLE32, чтобы понять можно ли выгрузить DLL

DllGetClassObject – юзает OLE32, чтобы получить компонент по переданному CLSID

1. **Назовите функцию COM-контейнера, которая вызывается OLE32 для получения указатель на фабрику классов.**

GetClassObject()

1. **Назовите функцию фабрики классов, в которой создается объект компонента.**

CreateInstance()

1. **Поясните назначение «счетчика экземпляров компонент». Где этот счетчик увеличивается и где уменьшается?**

Показывает количество активных экземпляров компонента. Изменяется в констукторе/деструкторе. Счетчик позволяет узнать можно ли выгружать Dll

1. **Назовите условие, при котором объект компонента удаляется.**

Когда количество ссылок на интерфейс становится равным нулю. Происходит это в Release – самоуничтожающийся объект

1. **Объясните на механизм блокировки COM-сервера (функция LockServer фабрики классов).**

Эту функцию вызывает клиент, передавая true || false, тем самым говорят лочит или нет он сервер. На сервере в зависимости от значения инкереметируется/декрементируется нужный счётчик. Далее в DllCanUnloadNow мы должны проверять этот счетчик на ноль -> если ноль, то никакой клиент не использует сервер.

CoCreateInstance

Создает и инициализирует по умолчанию один объект класса, связанный с указанным CLSID.

Вызовите **CoCreateInstance** , если вы хотите создать только один объект в локальной системе. Чтобы создать один объект в удаленной системе, вызовите функцию [CoCreateInstanceEx](https://learn.microsoft.com/en-us/windows/desktop/api/combaseapi/nf-combaseapi-cocreateinstanceex) . Чтобы создать несколько объектов на основе одного CLSID, вызовите функцию [CoGetClassObject](https://learn.microsoft.com/en-us/windows/desktop/api/combaseapi/nf-combaseapi-cogetclassobject) .

**Синтаксис**

С++Копировать

HRESULT CoCreateInstance(

[in] REFCLSID rclsid,

[in] LPUNKNOWN pUnkOuter,

[in] DWORD dwClsContext,

[in] REFIID riid,

[out] LPVOID \*ppv

);

**Параметры**

[in] rclsid

CLSID, связанный с данными и кодом, которые будут использоваться для создания объекта.

[in] pUnkOuter

Если **NULL** , это означает, что объект не создается как часть агрегата. Если не **NULL , указатель на интерфейс**[IUnknown](https://learn.microsoft.com/en-us/windows/desktop/api/unknwn/nn-unknwn-iunknown) агрегатного объекта (управляющий **IUnknown** ).

[in] dwClsContext

Контекст, в котором будет выполняться код, управляющий вновь созданным объектом. Значения берутся из перечисления [CLSCTX](https://learn.microsoft.com/en-us/windows/desktop/api/wtypesbase/ne-wtypesbase-clsctx) .

[in] riid

Ссылка на идентификатор интерфейса, который будет использоваться для связи с объектом.

[out] ppv

Адрес переменной указателя, которая получает указатель интерфейса, запрошенный в *riid* . После успешного возврата \* *ppv* содержит запрошенный указатель интерфейса. При ошибке \* *ppv* содержит **NULL** .

**Возвращаемое значение**

Эта функция может возвращать следующие значения.

|  |  |
| --- | --- |
| **Код возврата** | **Описание** |
| **S\_ОК** | Экземпляр указанного класса объектов успешно создан. |
| **REGDB\_E\_CLASSNOTREG** | Указанный класс не зарегистрирован в регистрационной базе данных. Также может указывать на то, что запрошенный вами тип сервера в перечислении [CLSCTX](https://learn.microsoft.com/en-us/windows/desktop/api/wtypesbase/ne-wtypesbase-clsctx) не зарегистрирован или значения для типов серверов в реестре повреждены. |
| **CLASS\_E\_NOAGGREGATION** | Этот класс нельзя создать как часть агрегата. |
| **E\_NOINTERFACE** | Указанный класс не реализует запрошенный интерфейс, или управляющий [IUnknown](https://learn.microsoft.com/en-us/windows/desktop/api/unknwn/nn-unknwn-iunknown) не предоставляет запрошенный интерфейс. |
| **E\_POINTER** | Параметр *ppv* имеет **значение NULL** . |

# Функция CoInitialize (objbase.h)

Инициализирует библиотеку COM в текущем потоке и определяет модель параллелизма как однопоточный апартамент (STA).

Новые приложения должны вызывать [CoInitializeEx](https://learn.microsoft.com/en-us/windows/desktop/api/combaseapi/nf-combaseapi-coinitializeex) вместо CoInitialize.

Если вы хотите использовать среду выполнения Windows, вы должны вместо этого вызвать [RoInitialize](https://learn.microsoft.com/en-us/windows/win32/api/roapi/nf-roapi-roinitialize) или [Windows::Foundation::Initialize](https://learn.microsoft.com/en-us/windows/win32/api/roapi/nf-roapi-initialize) .

## Синтаксис

HRESULT CoInitialize(

[in, optional] LPVOID pvReserved

);

## Параметры

[in, optional] pvReserved

Этот параметр зарезервирован и должен иметь значение **NULL** .

## Возвращаемое значение

Эта функция может возвращать стандартные возвращаемые значения E\_INVALIDARG, E\_OUTOFMEMORY и E\_UNEXPECTED, а также следующие значения.